True Costs of Cheap Labor Are Hard to Measure: Edge Deletion and VCG Payments in Graphs

Edith Elkind
Princeton University,
Department of Computer Science,
35 Olden St, Princeton, NJ 08544, USA
elkind@cs.princeton.edu

ABSTRACT

We address the problem of lowering the buyer's expected payments in shortest path auctions, where the buyer's goal is to purchase a path in a graph in which edges are owned by selfish agents. We show that by deleting some of the edges of the graph, one can reduce the total payment of the VCG mechanism by a factor of $\Theta(n)$. However, we prove that it is NP-hard to find the best subset of edges to delete, even if the edge costs are small integers, or the graph has very simple structure; in the former case, this problem is hard to approximate, too. On the positive side, we describe a pseudopolynomial time algorithm for series-parallel graphs and fixed edge costs. Also, we discuss the applicability of this algorithm for the case of general (probabilistic) costs and derive a general lower bound on the performance of algorithms that are based on expected edge costs.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems; G.2.2 [**Discrete Mathematics**]: Graph Theory

General Terms

Algorithms, Economics, Theory

Keywords

Shortest Paths, Auction, Mechanism Design

1. INTRODUCTION

The problem of purchasing a service from a team of independent contractors, and, in particular, its special case — buying an inexpensive path in a graph in which edges are owned by selfish agents — has recently received considerable attention [15, 7, 1, 6, 4, 12]. Its crucial difference from the ordinary shortest path problem is that the edge costs are assumed to be private, i.e., known to the owners of these edges only, and thus eliciting information about the actual costs is a nontrivial task. Any system of incentives that encourages the agents to reveal their costs to the auctioneer involves large payments to the participants: for this reason, in the worst case, any reasonable task allocation mechanism has to pay much more than the actual cost of the least expensive solution (see [1, 6]).

In light of these negative results, it has been proposed to shift the focus from worst case to average case considerations; indeed, in [6], it has been shown that if each agent's cost distribution is known, it it possible to design an optimal mechanism for this problem, i.e., one that minimizes the expected total payment. This mechanism proceeds by constructing a virtual cost function for each agent based on information about this agent's cost distribution and selecting the team with the smallest sum of virtual costs; the payments to agents depend on their reported costs as well as their virtual cost functions.

However, this approach has been justly criticized for assuming that the buyer has full knowledge of all agents' cost distributions; clearly, in many real-life situations this is not the case. Furthermore, when the mechanism is to be run frequently (as the case might be if this approach is used for allocating network resources, see [7]), the required virtual cost computations may be too expensive and time-consuming; a simple mechanism with reasonable payment properties would be more desirable.

One such candidate, first considered in this context by Nisan and Ronen [15], is the celebrated VCG mechanism ([17, 3, 8]), which works as follows: first, all edges submit their bids, then the buyer selects the cheapest path and pays each winning agent his threshold bid, i.e., the highest amount this agent can bid and still be on a winning path (for a more formal description, see Section 2).

The VCG mechanism is attractive from many perspectives. First, it is truthful and individually rational, i.e., each agent's best strategy is to announce his true cost, and if he does that, he is guaranteed that the payment he receives will cover his expenses. Second, the allocation and payments are easy to compute: the paper [9] proposes an algorithm for computing VCG payments whose running time is comparable to that of finding the shortest path in the underlying graph, and [7] shows how to efficiently compute VCG payments in a distributed fashion. Finally, this mechanism is detail-free, i.e., it does not make any assumptions about the underlying cost distributions. The combination of these features makes VCG a strong candidate for many applications.

Though, as mentioned above, the payments made by VCG (as well as any other mechanism) can be unacceptably high, this worst-case behavior is exhibited on a rather special class of graphs, namely, ones that have two fairly long vertex-disjoint paths. This suggests that the overall payment properties may be improved by modifying the underlying graph. Adding new edges to the graph may be beyond the capabilities of the mechanism designer, since this would require the designer to provide resources he does not have. However, deleting edges – prohibiting some agents from participating in the auction – is something that can easily be done.

This approach might appear counterintuitive at first, since edge deletion decreases the competition in the market. However, it is known to be useful in other domains, such as network design for selfish users, where a lot of work has been done on issues related to Braess's paradox (see [16] for a survey from a CS perspective, as well as many new results).

In this paper, we analyze the effects of edge deletion on the VCG payments. We consider the setting in which the mechanism designer knows the underlying graph as well as the edge cost distributions, though not the actual values of edge costs, and has to decide which edges to delete; after that, the mechanism elicits bids and runs VCG on the remaining graph. While this approach, too, requires the knowledge of the cost distributions, it is considerably more robust: small errors are unlikely to affect the set of edges that will be deleted. Moreover, the information about distributions is only used at the design step: after the optimal subgraph is selected, the mechanism can be run without referring to these distributions. This is useful if we view mechanism design as an offline process, while the actual execution of the mechanism takes place online: even though finding the best mechanism in this class may be difficult (and in this paper, we give an indication that this is likely be the case), using it requires very little computational effort.

Clearly, the very question of minimizing the expected payment is meaningless unless there is an underlying cost distribution: it can be shown that any mechanism that does not know anything about this distribution has an arbitrarily bad performance ratio with respect to a mechanism that uses this information. Therefore, we believe that our method strikes a reasonable balance between simplicity and performance; its computational efficiency has to be studied further, and in this paper, we make the first steps in analyzing it, both for general graphs and distributions and for interesting special cases.

We show that, indeed, edge deletion can be a useful tool: deleting a subset of edges may lower the expected payment by a factor of $\Omega(n)$, where n is the number of vertices of the graph; we also prove that this bound is tight. However, finding the best set of edges to delete can be difficult, and we prove a number of negative results that address this issue from different perspectives. First, we show that for general graphs, the problem is NP-hard even if all edge costs are small constants (i.e., the edge cost distributions are degenerate: the support of each distribution consists of a single point) as well as hard to approximate (specifically, we show that for undirected graphs, unless P = NP, the approximation ratio of any deterministic algorithm is at least n^{ε}; a similar result holds for directed graphs). An interesting class of graphs not covered by this hardness result is series-parallel graphs; however, if the edge costs are given in binary, we show that the problem is NP-hard even for this restricted class of graphs. We consider this result to be less devastating than the first one, as in practice, one can expect that the costs are not very large, in which case the problem can be solved efficiently: namely, we design an algorithm for series-parallel graphs (still under the constant edge cost assumption) whose running time is polynomial in n and C, where $C = \max\{c_i\}$, and c_i is the cost of edge e_i.

The constant cost assumption that is needed for this algorithm does not usually hold in practice; nevertheless, it can be the case that the cost of each edge is concentrated in a relatively small interval, in which case the algorithm (applied to expected edge costs) may still be useful. However, generalizing this approach to arbitrary distributions fails: it can be shown that any algorithm that operates on expected edge costs (rather than uses full information about cost distributions) has performance ratio of $\Omega(n^{1/4})$. The natural next step, then, is to analyze the performance of algorithms that know the first k moments of the distribution; the case of $k = 2$ is particularly interesting. We propose this as an intriguing open question.

1.1 Related Work

The private cost version of the shortest path problem was first introduced by [15], where the authors proposed to use the VCG mechanism and discussed the associated computational difficulties (later addressed by [9]). However, [15] did not attempt to minimize the payments to the agents. The first paper to raise this issue was [1], which showed that for a large class of mechanisms the worst-case payments must be large. Paper [6] generalizes the results of [1] to *all* dominant strategy mechanisms and designs the optimal Bayes-Nash mechanism for this problem.

Several papers [7, 6, 14, 12] compute the expected VCG payments for various graph models, both analytically and experimentally. In particular, in [14], it is shown that in many random graph models, the expected VCG payments are much lower than in the worst case scenarios of [1] and [6].

In a recent paper [4], Czumaj and Ronen discuss an interesting class of generalized VCG mechanisms, which includes the mechanisms considered in this paper, i.e., ones that ignore some of the edges. However, they do not address the related computational issues; in fact, it is not even clear that for a given graph, the optimal mechanism in this class will have a compact representation. Note that since the model of [4] is considerably richer than the one considered in this paper, our lower bounds do not imply any hardness results for their case.

A number of similar hardness results have already been known in the context of network design problems related to Braess's paradox [16], but none of these results apply directly to our scenario, and, moreover, our techniques are rather different from those used in [16].

The rest of this paper is organized as follows. Section 2 provides the background definitions and introduces some notation. Section 3 presents upper and lower bounds on the benefits of edge deletion. Section 4 contains the hardness results: Subsection 4.1 is devoted to hardness and inapproximability results for general graphs, while in Subsection 4.2, we show that the problem remains NP-hard when restricted to series-parallel graphs provided that edge costs are given in binary. Section 5 provides a counterpoint to these hardness results: it describes a pseudopolynomial algorithm for series-parallel graphs with constant edge costs. Section 6 discusses the difficulties that arise when we attempt to generalize the algorithm of Section 5 to probabilistic edge costs. We conclude in Section 7.

2. PRELIMINARIES

We model the network by a graph $G = (V, E)$, $|V| = n$, $|E| = m$, with two distinguished vertices s and t. Each edge $e_i \in E$ has an associated *cost* c_i, which is drawn at random from $\mathbb{R}^+$ according to a distribution F_i. We assume that these random choices are independent, and that the cost of each edge is private, that is, known to the owner of e_i only; the distributions F_i, however, may be known to the mechanism designer.

Some of the constructions in this paper use edge cost distributions whose support consists of a single point, i.e., we assume that the edge costs remain constant and the mechanism designer knows them; we refer to this setting as "fixed (constant) cost assumption". While this scenario appears ill-suited for VCG-like mechanisms (after all, under this assumption, the problem of minimizing the total payment has a trivial solution), using it to prove hardness results is completely legit: if a problem is hard in this simple setting, *a forteriori*, it is hard for general cost distributions.

The *cost* of a path P in G, which we denote by $|P|$, is the sum of the costs of the edges on the path. By the *shortest path* we mean the path that has the smallest cost; we use the terms 'shortest' and 'cheapest' interchangeably.

The costs that the edges announce for themselves are called *bids*. Since under the rules we are going to consider, truthful bidding is a dominant strategy, we can assume that all agents bid truthfully, i.e., the bid of the ith agent equals to his cost c_i. The auction mechanism is supposed to select a path between s and t; we refer to this path as the *winning* path, and say that edges on the selected path *win*, while edges not on the path *lose*.

Definition 1. A *mechanism* on a graph $G = (V, E)$ is a triple $(\mathcal{B}, Q(\mathbf{b}), M(\mathbf{b}))$, where $\mathcal{B} = (\mathbb{R}^+)^m$ is the set of possible bids, $Q : \mathcal{B} \mapsto [0, 1]^m$ is an *allocation rule*, and $M : \mathcal{B} \mapsto \mathbb{R}^m$ is a *payment rule*: $Q_i(\mathbf{b})$ is the probability that e_i is on the chosen path given that the bid vector is $\mathbf{b}$, and $M_i(\mathbf{b})$ is the corresponding payment to agent i.

We will also need the following notation: for any vector $\mathbf{v} \in \mathbb{R}^n$, define $\mathbf{v}_{-i} = (v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_n)$, and $(\mathbf{v}_{-i}, v') = (v_1, \ldots, v_{i-1}, v', v_{i+1}, \ldots, v_n)$.

Definition 2. An allocation rule is *monotone* if for any bid vector $\mathbf{b}$, any $i = 1, \ldots, m$, and any $b' > b_i$, we have $Q_i(\mathbf{b}_{-i}, b') \le Q_i(\mathbf{b})$, i.e., no edge can increase its probability of winning by raising its bid. Given a mechanism with a monotone allocation rule, the *threshold bid* of an edge e_i with respect to $\mathbf{b}_{-i}$ is a number b^t such that whenever all edges bid according to $(\mathbf{b}_{-i}, b')$, $b' < b^t$, e_i wins with probability 1, and whenever all edges bid according to $(\mathbf{b}_{-i}, b'')$, $b'' > b^t$, e_i loses with probability 1.

Definition 3. The Vickrey-Clarke-Groves (VCG) mechanism is the mechanism that selects a path with the lowest cost and pays each winning agent his threshold bid; the losing agents are paid 0. Given a graph G, $T_{\rm vcg}(G)$ denotes the expected total VCG payment on this graph; the cost distributions with respect to which it is computed will usually be clear from the context.

Observe that the VCG allocation rule is clearly monotone, and hence the notion of threshold bid is well-defined. Furthermore, the threshold bid of a winning edge can be interpreted as the sum of its actual bid and a bonus equal to the difference between the cost of the cheapest path that does not include this edge and the cost of the actual cheapest path.

We note that in economic literature, it is customary to stipulate that each bid (and, in particular, the threshold bid) should be feasible, i.e., belong to the support of the distribution of this bidder's costs. This restriction comes into play when the bidders' costs are known to be bounded, as it caps the payment to each edge by its maximum possible cost. However, this approach assumes that the support of each agent's cost distribution is known at runtime, which conflicts with the goal of having a detail-free mechanism. Hence, in this paper we dispense with this requirement.

3. EFFECTS OF EDGE DELETION: UPPER AND LOWER BOUNDS

We start by showing that edge deletion can have a dramatic effect on expected payments, even if we restrict attention to graphs with unit edge costs. An easy argument sketched below shows that edge deletion cannot reduce the expected payment by more than a factor of n; we prove a matching lower bound of $n - o(n)$ for general costs and $\Theta(n)$ for unit costs. This provides motivation for looking into algorithmic properties of edge deletion, which is done in subsequent sections.

While conceptually easy, the examples used in these proofs give us a glimpse of the interplay between the three factors that affect VCG payments in networks: the actual cost of the winning path, the difference between this cost and the cost of the second cheapest path, and the number of edges on the winning path.

CLAIM 1. *Let G be a graph with fixed edge costs and let l be the number of edges on the shortest s-t path P in G. Then for any $G' \subseteq G$ we have $T_{vcg}(G)/T_{vcg}(G') \le l$.*

PROOF. For each $e \in P$, let t_e be the cost of the cheapest path in $G \setminus \{e\}$. Set $e_0 = \text{argmax}_{e \in P}\, t_e$. Clearly, $T_{\rm vcg}(G) = |P| + \sum_{e \in P}(t_e - |P|) \le l t_{e_0}$. Now, consider an arbitrary $G' \subseteq G$, and let P' be the shortest path in G'. If $e_0 \notin P'$, we have $|P'| \ge t_{e_0}$. Otherwise, e_0 is a winning edge; the length of the shortest path in $G' \setminus \{e_0\}$ is at least t_{e_0}, so the bonus paid to e_0 is at least $t_{e_0} - |P'|$, and hence the total payment is at least $|P'| + (t_{e_0} - |P'|)$. In both cases, $T_{\rm vcg}(G')$ is at least $t_{e_0} \ge T_{\rm vcg}(G)/l$. $\square$

REMARK 1. *This argument can be generalized to probabilistic edge costs by considering an edge e_0' that maximizes the expected length of the shortest path in $G \setminus \{e_0'\}$; by linearity of expectation, $T_{vcg}(G)$ can then be bounded by $nt_{e_0'}$ and hence $T_{vcg}(G)/T_{vcg}(G') \le n$; we omit the details.*

REMARK 2. *One can interpret Claim 1 as saying that the trivial algorithm that deletes no edges provides an l-approximation to our problem. In Section 4, we show that one cannot expect to design a polynomial-time algorithm*

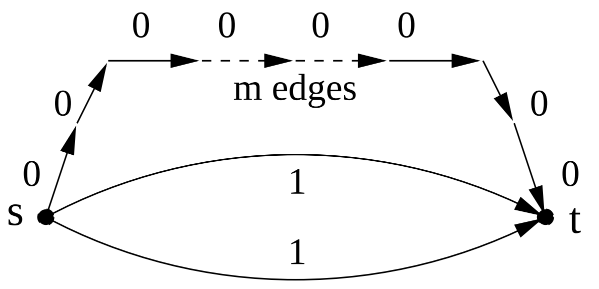

Figure 1: Arbitrary weights: edge deletion reduces total VCG payment by a factor of $m - o(m)$.

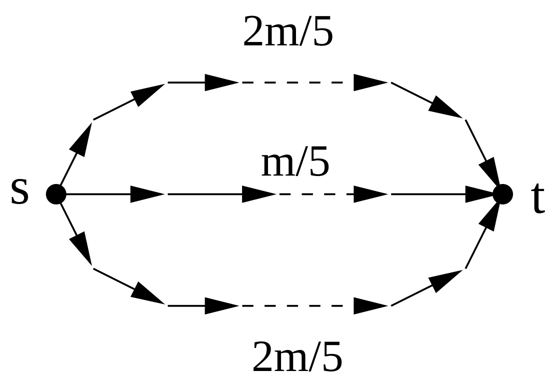

Figure 2: Unit weights: edge deletion reduces total VCG payment by a factor of $\Omega(m)$.

with a significantly better approximation ratio, at least for general graphs.

Claim 2. *Deleting a subset of edges can decrease the expected VCG payments in a network by a factor of $(1 - o(1))m$, where m is the number of edges in the original network. Moreover, even among graphs with unit edge costs, there exists a graph G such that for some $G' \subseteq G$ we have $Tvcg(G)/Tvcg(G') = \Omega(m)$.*

Proof. To prove the first statement of the theorem, consider a graph that consists of an $(m-2)$-edge path between s and t all edges of which have cost 0, and two parallel edges between s and t of cost 1 each. If no edges are deleted, the $(m-2)$-edge path wins, and each of its edges has to be paid its threshold bid 1; on the other hand, if the $(m-2)$-edge path is destroyed, one of the remaining two edges will be declared the winner, and its threshold bid (and hence the total payment) will be 1 as well. Hence, deleting any edge on the $(m-2)$-edge path lowers the payments from $m-2$ to 1, i.e., by a factor of $(1 - o(1))m$.

The second statement is proved in a similar manner: this time, the network in question consists of three disjoint s-t paths; one of these paths has $m/5$ edges, and the other two have $2m/5$ edges each. The cost of each edge is 1. Clearly, in the original graph the shorter path wins, and each edge on this path is paid its cost plus a bonus of $m/5$, so the total payment is $m/5(m/5+1)$. Deleting any edge on the shorter path causes one of the longer paths to win, in which case the VCG payment drops to $2m/5$. $\square$

4. HARDNESS RESULTS

4.1 General Graphs

While in the example above, there were obvious candidates for deletion, the following theorem shows that the situation is not always so simple, even for graphs with fixed edge costs.

Formally, we define the following problem:
Min VCG Payment
Given a network $\Gamma = (G = (V, E), s, t)$, $|V| = n$, $|E| = m$, a vector $(c_1, \dots c_m)$, where $c_i \in \mathbb{Z}^+$ is the cost of the ith edge, and an integer T, decide whether there is a subset E_0 of E such that the expected payment of the VCG mechanism on $G_0 = (V, E_0)$ is at most T.

Theorem 1. Min VCG Payment *is NP-complete even if $c_i = 1$ for $i = 1, \dots, m$.*

Proof. It is easy to see that this problem is in NP: for a given subset of edges E_0, we can compute the VCG payment on (V, E_0) and compare it with T.

The NP-hardness is proved by reduction from Longest Path. An instance of Longest Path consists of an unweighted graph $G = (V, E)$ and a target l; the goal is to decide if G contains a simple path of length at least l. For our purposes, we will need a modified version of this problem, in which we are also given two distinguished vertices $s, t \in V$ and would like to know if there is a path from s to t in G of length exactly k. We refer to this problem as Exact Longest Path. It is easy to see that an oracle for Exact Longest Path allows us to solve the original problem in polynomial time (and hence Exact Longest Path is NP-hard, too): one can call this oracle for all possible choices of s, t, and k, $l \le k \le n$.

Given an instance $(G = (V, E), s, t, k)$ of Exact Longest Path, the instance of Min VCG Payment is constructed by adding a new sink t', an n-edge path P_1 that connects t and t', and an $(n+k)$-edge path P_2 that connects s and t'. The cost of each edge is set to 1, and $T = n + k$. The resulting instance $(V', E' = E \cup P_1 \cup P_2, s' = s, t', T)$ of Min VCG Payment is shown in Figure 3.

Suppose that we are given a 'yes'-instance of the Exact Longest Path problem, i.e., G contains an s-t path P, $|P| = k$. If we delete all edges in $E \setminus P$ from G', the resulting graph would consist of two vertex-disjoint paths of cost $n+k$ each, so the total VCG payment will be $n+k$, and therefore we have a 'yes'-instance of Min VCG Payment.

Conversely, suppose that we are given a 'no'-instance of Exact Longest Path. Fix a subset $E'_0 \subseteq E'$. The shortest s-t path in $(V, E \cap E'_0)$ has length k'; since there is no s-t path of length k in G, either $k' < k$ or $k' > k$. In the former case, the shortest path has length $k' + n$ and is contained in the upper part of the graph; in particular, all edges of P_1 are on the winning path. Each of these edges is paid $1 + (k+n) - (k'+n) \ge 2$, so the total payment is at least $2n > n + k$ (since we are looking for a simple path, we can assume $k < n$). In the latter case, the shortest path is P_2; similarly, each edge on P_2 is paid its cost plus a bonus of size $(k'+n) - (k+n)$, so the total payment is at least $2(k+n) > k + n$. Therefore, for any such E'_0, the total payment is greater than $n + k$, so we get a 'no'-instance of Min VCG Payment. $\square$

One of the natural approaches to coping with NP-hardness is to focus on approximation algorithms for the problem at

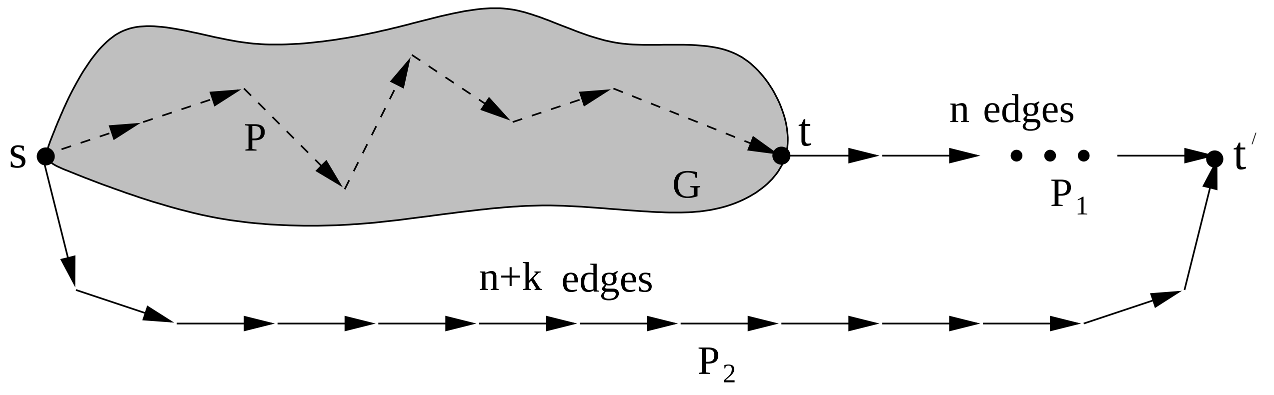

Figure 3: Reduction from Longest Path.

hand. However, for Min VCG Payment, this technique is not very productive: the next theorem demonstrates that a polynomial-time algorithm with a reasonable approximation ratio for Min VCG Payment is unlikely to exist. This is hardly surprising, since Longest Path is known to be hard to approximate. In particular, paper [11] shows that for undirected graphs, no polynomial time algorithm can approximate the length of the longest path within $2^{\log^{1-\varepsilon} n}$ for any $\varepsilon > 0$ unless $NP \subset DTIME(2^{\log^{O(1/\varepsilon)} n})$. For directed graphs, a recent paper of Bjorklund et al. [2] proves a strong hardness result under more standard assumptions: unless $P \neq NP$, Longest Path cannot be polynomial time approximated within $n^{1-\varepsilon}$. We show that this translates into similar inapproximability results for Min VCG Payment as well. Our reduction applies to both directed and undirected case; therefore, instead of treating these two cases separately, we prove a general theorem that describes the relationship between inapproximability ratios for these problems.

Theorem 2. *Consider the optimization version of* Min VCG Payment*, where the goal is to find a subset of edges that minimizes total VCG payment. For any $\alpha > 2$, given a polynomial-time algorithm for this problem whose approximation ratio is α, one can construct a polynomial-time algorithm for* Longest Path *that has approximation ratio at most α.*

Proof. We show how to construct an α-approximation algorithm for Longest Path using an oracle for Min VCG Payment that is guaranteed to produce an α-approximation to the optimal solution. The input to our algorithm is $G = (V, E)$, and the goal is to find a path of length at least l/α, where l is the length of the longest simple path in G.

We use the construction described in the proof of Theorem 1; namely, for all $s, t \in V$ and any k, $k = 1, \dots, n$, we consider an instance of Exact Longest Path with source s, sink t, and target length k, transform it into an instance of Min VCG Payment as described above, and call our oracle on this instance. The oracle returns a graph that necessarily contains an s-t path (otherwise, the VCG payments would be infinite); pick the shortest of them. This procedure results in a list of n^3 s-t paths in the original graph. Now, check if G has a path of length at least 4; if yes, add this path to the list, and return the longest path in this list. It remains to show that this algorithm is an α-approximation for the Longest Path problem.

Suppose that the longest path in G has length $\bar{l}$; let $\bar{s}$ and $\bar{t}$ be the first and the last vertex on this path, respectively. Consider the instance of Min VCG Payment that corresponds to $((V, E), \bar{s}, \bar{t}, \bar{l})$. Obviously, one can achieve the total payment of $\bar{l} + n$ by deleting all edges of G that are not on this path; therefore, the oracle returns a graph G' on which total VCG payment is at most $\alpha(\bar{l} + n) \leq 2\alpha n$. The length of the shortest s-t' path P in $G \cup P_1$ (for definition of P_1, see the proof of Theorem 1) is at most $\bar{l} + n$. If it is exactly equal to $\bar{l} + n$, we are done, since we have identified a path of length $\bar{l}$ in G. Otherwise, all edges on P_1 are on the winning path. Each of these n edges has the same threshold bid; therefore, each of them is paid at most 2α: otherwise, the total payment is at least $2\alpha n$. On the other hand, the threshold bid of any $e \in P_1$ is equal to $1 + |P_2| - (|P| + |P_1|) = 1 + (\bar{l} + n) - (|P| + n)$. We have $1 + \bar{l} - |P| \leq 2\alpha$, or $|P| \geq \bar{l} - 2\alpha$.

Now, if $\bar{l} \leq 4$, our algorithm produces an exact solution. If $4 < \bar{l} < 4\alpha$, then a path of length at least 4 that was found in the last step of the algorithm is an α-approximation. If $\bar{l} > 4\alpha$, we have $\bar{l} - 2\alpha > \bar{l}/2$, so P is a 2-approximation. In all cases, we have produced an α-approximation to Longest Path. □

Remark 3. *The proof of Theorem 2 implies a stronger inapproximability result than stated in the theorem; however, our aim is not to prove the strongest possible bound, but rather to decide whether there is a polynomial-time algorithm with a reasonable approximation guarantee. Showing that our problem is at least as hard to approximate as* Longest Path *fulfills this purpose.*

4.2 Series-Parallel Graphs

Another classical approach to tackle NP-hardness is to consider special cases of the problem. In our case, it is natural to concentrate on classes of graphs that admit efficient algorithms for finding a longest path. However, it turns out that Min VCG Payment is hard even for very simple graphs, for which finding a longest path is trivial, such as series-parallel graphs (for the definition, see Section 5) This time, the source of hardness is in the edge costs rather than in the structure of the graph: if the costs are given in binary, it may be hard to find a path that has a prescribed cost, even in a simple graph.

Theorem 3. Min VCG Payment *is NP-hard even if $G = (V, E)$ is a series-parallel graph.*

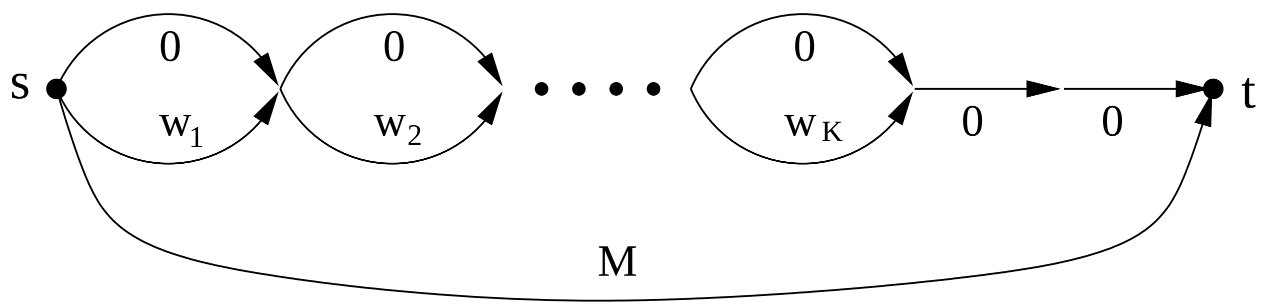

Figure 4: Hard series-parallel graph.

PROOF. The theorem is proved by constructing a reduction from SUBSET SUM. An instance of SUBSET SUM consists of a list of positive integers $(w_1, \dots, w_K)$ and a target M; the goal is to decide whether there exist a subset I of $\{1, \dots, K\}$ such that $\sum_{i \in I} w_i = M$.

Given an instance of SUBSET SUM, we construct an instance of MIN VCG PAYMENT as follows. The graph $G = (V, E)$ has $K+3$ vertices and $2K+3$ edges. For each pair of vertices (v_i, v_{i+1}), $i = 0, \dots, K-1$, there are two parallel edges e_{2i} and e_{2i+1} between these two vertices; we set $c_{2i} = 0$, $c_{2i+1} = w_i$. Also, there are edges $e_{2K} = (v_{2K}, v_{2K+1})$ and $e_{2K+1} = (v_{2K+1}, v_{2K+2})$, which have cost 0, and an edge $e_{2K+2} = (v_0, v_{2K+2})$, which has cost M. We set $s = v_0$, $t = v_{2K+2}$. The resulting graph G is depicted in Fig. 4; clearly, it belongs to the family of series-parallel graphs. The target payment T is set to be equal to M.

Observe that if no edges are deleted, VCG will always choose the path of cost 0 that consists of all edges of cost 0, i.e., e_{2i}, $i = 0, \dots, K-1$, e_{2K}, and e_{2K+1}. As the threshold bids of both e_{2K} and e_{2K+1} are equal to M, the total payment is at least $2M$. Intuitively, to reduce the payment, we would like to lower the threshold bids of these edges; this can be achieved by closing the gap between the cost of the shortest path in the upper part of the graph and the cost of the edge e_{2K+2}, i.e., M.

First, we claim that if we start with a 'yes'-instance of SUBSET SUM, then by deleting some edges from G, we can obtain a graph $G' = (V, E')$ such that the length of the shortest path in the upper part of G', i.e. in $(V, E' \setminus \{e_{2K+2}\})$, is exactly M. Indeed, suppose that there exists a set $I \subseteq \{1, \dots, K\}$ such that $\sum_{i \in I} w_i = M$. Then we can set $E' = E \setminus \{e_{2i} \mid i \in I\}$.

For this G', the payment of the VCG mechanism is M: there are two vertex-disjoint paths of total cost M (the path in $(V, E' \setminus \{e_{2K+2}\})$ and the edge e_{2K+2}), and no path that is cheaper than M. Hence, no matter which of these two paths is declared the winner, none of the edges on the winning path can raise its bid without losing the auction, so each of them is paid exactly its bid. Therefore, a 'yes'-instance of SUBSET SUM corresponds to a 'yes'-instance of MIN VCG PAYMENT.

Conversely, suppose that we start with a 'no'-instance of SUBSET SUM. It cannot be the case that for some $E' \subseteq E$, the cost of the cheapest path from s to t in $(V, E' \setminus \{e_{2K+2}\})$ is M, because otherwise the set I that provides a solution for SUBSET SUM can be read off G': consider the set of deleted edges of cost 0, i.e., $S = (E \setminus E') \cap \{e_{2i} \mid i = 0, \dots, K-1\}$ (clearly, if the cheapest path in $(V, E' \setminus \{e_{2K+2}\})$ has finite cost, e_{2K} and e_{2K+1} have not been deleted) and set $I = \{i \mid e_{2i} \in S\}$.

Now, consider a graph $G'' = (V, E'')$, $E'' \subseteq E$, and suppose that the cost of the cheapest path from s to t in $(V, E'' \setminus \{e_{2K+2}\})$ is A, $A \neq M$. If $A > M$, then e_{2K+2} wins the auction and is paid A; if $A < M$, the threshold bids of both e_{2K} and e_{2K+1} are equal to $M - A$, so the total payment is at least $2M - A > M$.

Hence, in this case the total payment of VCG on any subgraph of G is at least $M+1$, and thus we have obtained a 'no'-instance of MIN VCG PAYMENT. □

REMARK 4. *Our hardness results use graphs with fixed edge costs, but it is easy to see that they hold for atomless cost distributions as well. In this case, the problem instance must include a compact description of the edge cost distributions (e.g., if the cost of edge e_i is drawn uniformly from $[a_i, b_i]$, $i = 1, \dots, m$, the problem description should consist of the graph G, m pairs (a_i, b_i), and a target T). The corresponding hard instance can then be created by replacing the fixed costs $w_1, \dots, w_n$ in the proof of Theorem 3 with uniform distributions $U[w_1 - \varepsilon, w_1 + \varepsilon], \dots, U[w_n - \varepsilon, w_n + \varepsilon]$ for sufficiently small ε; a similar modification works for Theorem 1.*

However, we cannot claim that this more general version of MIN VCG PAYMENT *is NP-complete: it is unclear how to efficiently compute the expected VCG payment for a given graph when edge costs are not constant. However, if the corresponding distributions are efficiently samplable and satisfy some reasonable regularity conditions, one can use Monte Carlo method to estimate the expected payments.*

REMARK 5. *A related, but seemingly easier question, is whether an input graph can benefit at all from edge deletion, i.e., whether there is a subset of edges E' such that the VCG payments on $(V, E \setminus E')$ are smaller than on the original graph. It turns out that the construction in the proof of the previous theorem can be modified to show that this problem is NP-hard, too; we omit the details.*

5. ALGORITHM FOR SERIES-PARALLEL NETWORKS

In this section, we describe an algorithm that solves MIN VCG PAYMENT on series-parallel networks with fixed edge costs. The running time of this algorithm is polynomial in C, n, and m, where $C = \max\{c_1, \dots, c_m\}$.

The class of series-parallel networks can be inductively defined as follows:

Definition 4. A *series-parallel network* (SPN) is a network (V, E, s, t), such that one of the following conditions holds:

- **Base case:** A single edge (s,t), i.e., a network of the form $(\{s,t\},\{(s,t)\},s,t)$ is an SPN.

- **Series:** Suppose that $\Gamma_1=(V_1,E_1,s_1,t_1)$ and $\Gamma_2=(V_2,E_2,s_2,t_2)$ are SPN such that $V_1\cap V_2=\emptyset$. Set $V=V_1\cup V_2$, $E=E_1\cup E_2$, and merge t_1 with s_2. Then $(V,E,s=s_1,t=t_2)$ is an SPN.

- **Parallel:** Suppose that $\Gamma_1=(V_1,E_1,s_1,t_1)$ and $\Gamma_2=(V_2,E_2,s_2,t_2)$ are SPN such that $V_1\cap V_2=\emptyset$. Set $V=V_1\cup V_2$, $E=E_1\cup E_2$, and merge s_1 with s_2 and t_1 with t_2. Then $(V,E,s=s_1=s_2,t=t_1=t_2)$ is an SPN.

Our algorithm for Min VCG Payment uses dynamic programming approach. It is based on a subroutine that, given a network $\Gamma=(V,E,s,t)$, which has been obtained from lower-level networks Γ_1 and Γ_2 by serial or parallel composition, computes a family of candidate solutions $\mathcal{F}(\Gamma)=\{G^{i,k}\}_{0\le i\le k\le nC}$ by recursively computing the corresponding families for component networks Γ_1 and Γ_2 and then combining them; if Γ is a single-edge network, $\mathcal{F}(\Gamma)$ is computed from scratch. The important property of $\mathcal{F}(\Gamma)$ is that it is guaranteed to contain the optimal solution; therefore, when the subroutine returns $\mathcal{F}(\Gamma)$ for the input network Γ, the algorithm simply finds the element of $\mathcal{F}(\Gamma)$ that has the smallest total VCG payment.

The graph $G^{i,k}=(V,E^{i,k})$ for a network $\Gamma=(V,E,s,t)$ is defined as follows. Consider the network Γ' that is obtained from Γ by adding a new s-t edge $e^{(k)}$ of cost k. Consider all edge subsets $E'\subseteq E\cup\{e^{(k)}\}$ such that the length of the shortest s-t path in (V,E') is i. If no such E' exists, i.e., there is no s-t path of cost i in Γ, set $E^{i,k}=\emptyset$. Otherwise, among all such E', pick the one that minimizes the total VCG payment on (V,E'); denote it by $\bar{E}$, and set $E^{i,k}=\bar{E}\setminus e^{(k)}$. In other words, $G^{i,k}$ is the subgraph of (V,E) that minimizes the VCG payments assuming that the shortest path is required to have length i and the bonus to each edge is bounded by $k-i$. We denote the total VCG payment on $(V,E^{i,k}\cup\{e^{(k)}\})$ by $T^{i,k}$.

Obviously, if the network in question is 2-connected, then an optimal solution to the optimization version of Min VCG Payment is given by one of the $E^{i,nC}$, $i=0,\dots,nC$: the auxiliary edge $e^{(nC)}$ is too expensive to be of any use. It remains to show how to recursively compute $G^{i,k}$. When Γ is a one-edge network, the computation is trivial. In what follows, we show how to construct $G^{i,k}$ for a network Γ obtained from Γ_1 and Γ_2 by serial or parallel connection, assuming that the corresponding families $G_1^{i,k}$ and $G_2^{i,k}$ for Γ_1 and Γ_2 have already been constructed.

5.1 Serial Connection

Suppose that Γ has been obtained by connecting Γ_1 and Γ_2 in series, and our goal is to find $G^{i,k}$. To do that, we prove two claims that relate the value of the optimal solution for Γ to the corresponding values for Γ_1 and Γ_2.

Let $e^{(k)}$ be an s-t edge of cost k, let $e_1^{(k-i+j)}$ be an s_1-t_1 edge of cost $k-(i-j)$, and let $e_2^{(k-j)}$ be an s_2-t_2 edge of cost $k-j$.

Claim 3. *For any j, $0\le j\le i$, and any choice of optimal subgraphs $G_1^{j,k-(i-j)}$, $G_2^{i-j,k-j}$,*

$$Tvcg(V,E_1^{j,k-(i-j)}\cup E_2^{i-j,k-j}\cup\{e^{(k)}\})\le T_1^{j,k-(i-j)}+T_2^{i-j,k-j}.$$

Claim 4. *There exists a j, $0\le j\le i$, such that*

$$T_1^{j,k-(i-j)}+T_2^{i-j,k-j}\le T^{i,k}.$$

These claims imply we can find $G^{i,k}$ by going over all possible pairs $(G_1^{j,k-(i-j)},G_2^{i-j,k-j})$, picking the one with the smallest sum of VCG payments, and setting $E^{i,k}=E_1^{j,k-(i-j)}\cup E_2^{i-j,k-j}$. By Claim 3, if we do that, we are guaranteed to spend as little as $T_1^{j,k-(i-j)}+T_2^{i-j,k-j}$, and by Claim 4, we cannot expect to pay less than that.

Proof (of Claim 3). Fix j and subgraphs $G_1^{j,k-(i-j)}$, $G_2^{i-j,k-j}$, and set

$$\hat{E}=E_1^{j,k-(i-j)}\cup E_2^{i-j,k-j}.$$

Suppose that P_1 is a shortest path in $G_1^{j,k-(i-j)}$ and P_2 is a shortest path in $G_2^{i-j,k-j}$. Then $P=P_1\cup P_2$ is a shortest path in $(V,\hat{E})$ and $|P|=|P_1|+|P_2|=i$. We show that the VCG payments to edges on P in $(V,\hat{E}\cup\{e^{(k)}\})$ do not exceed the payments to these edges in $(V_1,E_1^{j,k-(i-j)}\cup\{e_1^{(k-i+j)}\})$ and $(V_2,E_2^{i-j,k-j}\cup\{e_2^{(k-j)}\})$, respectively.

Consider an arbitrary edge $e\in P_1$. If the shortest path in $(V_1,E_1^{j,k-(i-j)}\cup\{e_1^{(k-i+j)}\}\setminus\{e\})$ is $e_1^{(k-i+j)}$, then in this graph, e is paid its cost plus a bonus of $k-(i-j)-j=k-i$. In $(V,\hat{E}\cup\{e^{(k)}\})$, e is on a path of cost i, and there is a path of cost k (namely, $e^{(k)}$) that does not include e, so e will be paid at most its cost plus a bonus of $k-i$, i.e., no more than in $(V_1,E_1^{j,k-(i-j)}\cup\{e_1^{(k-i+j)}\})$. On the other hand, suppose that the shortest path in $(V_1,E_1^{j,k-(i-j)}\cup\{e_1^{(k-i+j)}\}\setminus\{e\})$ does not use $e_1^{(k-i+j)}$. Denote this path by P_e; the payment to e in this case is $|P_e|-j$. By construction, $\hat{E}$ contains the path $P_e\cup P_2$. Therefore, in $(V,\hat{E})$, e is paid at most its cost plus a bonus of $|P_e|+|P_2|-i=|P_e|+(i-j)-i=|P_e|-j$.

A similar argument applies to any $e\in P_2$, which proves the claim. □

Proof (of Claim 4). Suppose that P is a shortest path in $G^{i,k}$, and set $P_1=P\cap E_1$, $P_2=P\cap E_2$. Suppose also that $|P_1|=j$, and hence $|P_2|=i-j$. We will prove the inequality $T_1^{j,k-(i-j)}+T_2^{i-j,k-j}\le T^{i,k}$ for this particular value of j.

Set $\hat{E}_1=E^{i,k}\cap E_1$, $\hat{E}_2=E^{i,k}\cap E_2$. Observe that P_1 is the shortest s_1-t_1 path in $(V_1,\hat{E}_1)$, so the total VCG payment in $(V_1,\hat{E}_1\cup\{e_1^{(k-i+j)}\})$ is the sum of payments to all edges on P_1 in this graph.

Consider an arbitrary edge $e\in P_1$. We show that the payment to e in $(V_1,\hat{E}_1\cup\{e_1^{(k-i+j)}\})$ does not exceed the payment to e in $(V,E^{i,k}\cup\{e^{(k)}\})$. Indeed, we have one of the following cases. If the shortest path in $(V,E^{i,k}\cup\{e^{(k)}\}\setminus\{e\})$ is $e^{(k)}$, then in $(V,E^{i,k}\cup\{e^{(k)}\})$, e is paid its cost plus a bonus of $k-i$. In $(V_1,\hat{E}_1\cup\{e_1^{(k-i+j)}\})$, e is on a path of cost j, and there is a path of cost $k-(i-j)$ (namely, $e_1^{(k-i+j)}$) that does not include e, so e will be paid at most its cost plus a bonus of $k-(i-j)-j=k-i$, i.e., no more than in $(V,E^{i,k}\cup\{e^{(k)}\})$. On the other hand, if the shortest path in $(V,E^{i,k}\cup\{e^{(k)}\}\setminus\{e\})$ does not use $e^{(k)}$, then this path is the union of P_e, which is the shortest s_1-t_1 path in $(V,\hat{E}_1\setminus\{e\})$ and P_2, which is the shortest s_2-t_2 path in $G_2^{i-j,k-j}$. Therefore, in $(V,E^{i,k}\cup\{e^{(k)}\})$, e is paid its cost plus a bonus of $|P_e|+|P_2|-|P|=|P_e|-j$. By construction,

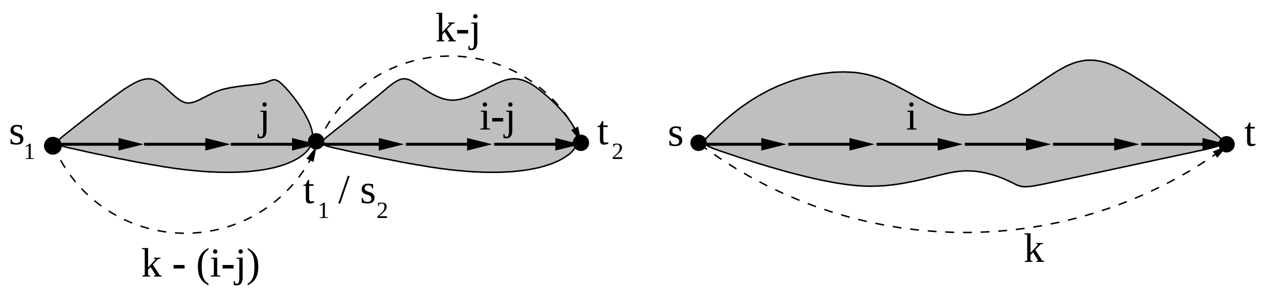

Figure 5: Series connection.

the graph $(V_1, \hat{E}_1)$ contains the path P_e, so in this case, the payment to e in $(V_1, \hat{E}_1 \cup \{e_1^{(k-i+j)}\})$ is at most its cost plus a bonus of $|P_e| - j$.

Similarly, the VCG payment to an arbitrary edge on P_2 in $(V_2, \hat{E}_2 \cup \{e_2^{(k-j)}\})$ does not exceed the payment to this edge in $(V, E^{i,k} \cup \{e^{(k)}\})$. By optimality of $G_1^{j,k-(i-j)}$ and $G_2^{i-j,k-j}$, the payments that are produced by using $E_1^{j,k-(i-j)}$ and $E_2^{i-j,k-i}$ in place of $\hat{E}_1$ and $\hat{E}_2$ can only be smaller, so from that, the result follows. □

5.2 Parallel Connection

Now suppose that Γ has been obtained by connecting Γ_1 and Γ_2 in parallel, and let P be the shortest path in $G^{i,k}$. Obviously, P has length i and lies entirely in Γ_1 or Γ_2. Without loss of generality, assume $P \subseteq E_1$. Then the shortest path P' in $G^{i,k} \cap \Gamma_2$ has length at least i, and all edges of $G^{i,k} \cap \Gamma_2$ that do not lie on P' can be deleted without affecting the total payment. Moreover, P' itself is only used to compute the VCG payments to edges on P, so any two paths in $G^{i,k} \cap \Gamma_2$ that have the same cost will result in identical total VCG payments, and it does not matter which one we pick. Finally, if it is known that $|P'| = j$, $i \le j < k$, we are in exactly the same setting as when selecting $G_1^{i,j}$.

This suggests that we can find $G^{i,k}$ as follows: for $j = i, \dots, k$, let Q_1^j be the shortest s-t path in $G_1^{j,k}$ (obviously, $|Q_j^1| = j$) if the latter is nonempty and $\emptyset$ otherwise; Q_2^j is defined similarly. Let e' be an s-t edge of cost k. Set $G_1^{i,j,k} = (V, E_1^{i,j} \cup Q_2^j \cup \{e'\})$, $G_2^{i,j,k} = (V, E_2^{i,j} \cup Q_1^j \cup \{e'\})$. Among these graphs, pick the one that minimizes total VCG payments, denote it by $G^{i,j,k} = (V, E^{i,j,k})$, and set $E^{i,k} = E^{i,j,k} \setminus \{e'\}$.

6. GENERAL DISTRIBUTIONS OF EDGE COSTS

The algorithm described in the previous section is designed for the case when for each edge, the support of its cost distribution consists of a single point. In practice, there is usually some uncertainty about the edge costs. However, if the cost of each edge is known to lie within a small interval, so that this uncertainty has little effect on relative costs of different paths, our algorithm (applied to expected edge costs) may still provide a solution of acceptable quality.

However, the situation changes dramatically when we consider distributions that have significant variance. Not only is it the case that the algorithm of Section 5 does not perform very well, but *any* algorithm that operates on expected edge costs rather than uses a full description of edge cost distributions is essentially useless.

More formally, we can prove the following theorem.

THEOREM 4. *Let $G = (V, E)$ be a graph, and let $D = (D_1, \dots, D_m)$ be a vector of edge cost distributions. For $i = 1, \dots, m$, let $c_i = \mathbf{E}[D_i]$ be the expected cost of the ith edge. Let A be an arbitrary algorithm that given a graph G and a vector of expected edge costs $c = (c_1, \dots, c_m)$ selects a subset $E' \subset E$. Let E'' be the subset of edges such that $Pvcg(V, E'') = \min\{Pvcg(V, E_0) \mid E_0 \subseteq E\}$. Then for any n_0 there is a graph G with $n > n_0$ vertices and a vector D such that $Pvcg(V, A(G, c))/Pvcg(V, E'') = \Omega(n^{1/4})$. Moreover, D can be chosen so that each D_i is either the Bernoulli distribution with mean $1/2$ (i.e., $P(0) = P(1) = 1/2$), or the trivial distribution with the same mean (i.e., $P(1/2) = 1$).*

PROOF. Consider a graph G that consists of 4 vertex disjoint paths from s to t: P_1 has $m^{5/8}$ edges, P_2 has $3m^{5/8}$ edges, P_3 and P_4 have $m/2 - 2m^{5/8}$ edges each. Consider the following two cases:

1. The cost of each edge is $1/2$. In this case, if we delete P_1 and P_2, the total payment is exactly $m/2 - 2m^{5/8}$. On the other hand, if either P_1 or P_2 remains in the graph, the gap B between the lengths of the shortest path and the second-shortest path is at least $m^{5/8}$; since there are at least $m^{5/8}$ edges on the winning path, and each of them is paid at least B, the total payment is at least $m^{5/4}$.

2. The cost of each edge is distributed as $B(1/2)$: $P(0) = P(1) = 1/2$. In this case, we have $\mathbf{E}[c(P_2) - c(P_1)] = \Theta(m^{5/8})$, $\mathbf{E}[c(P_3) - c(P_2)] = \Theta(m)$, $\mathbf{E}[|c(P_4) - c(P_3)|] = \Theta(\sqrt{m})$. Consequently, if no edges are removed, the expected payment is $\Theta(m^{5/4})$. If exactly one of P_1 and P_2 is removed, the expected gap B between the lengths of the shortest path and the second-shortest path (and hence the bonus paid to each winning agent) is $\Theta(m)$, and there are $\Theta(m^{5/8})$ agents who receive this bonus, so the expected payment is $\Theta(m^{13/8})$. If both P_1 and P_2 are removed, we have $B = \Theta(\sqrt{m})$, and the expected payment is $mB = \Theta(m^{3/2})$.

In both cases, the expected payment under the optimal solution differs from the expected payment under any other candidate solution by a factor of $\Omega(m^{1/4})$. However, the optimal solutions in these two cases are different, even though an algorithm that operates on expected edge costs cannot distinguish between these two settings. □

7. CONCLUSIONS AND FUTURE WORK

We introduced a new class of VCG-based mechanisms for the shortest path problem: a mechanism in this class is obtained by deleting some of the edges of the underlying graph and running VCG on the remaining set of edges. Once designed, any such mechanism can be run efficiently and in a detail-free manner; furthermore, the best mechanism in this class can substantially lower the expenses of the buyer compared to the traditional VCG mechanism.

However, it is NP-hard to determine what is the optimal set of edges to delete, or even if there is a set of edges whose deletion is beneficial to the buyer; moreover, for general graphs the problem is hard to approximate as well. The problem remains hard when restricted to series-parallel graphs, but can be solved efficiently if we additionally require that all edge costs are small constants. It would be interesting to identify other classes of graphs for which finding the optimal set of edges to delete is easy.

Another set of questions is related to graphs with non-constant edge costs. In particular, we would like to know whether it is possible to design an efficient deterministic algorithm that computes the expected VCG payments when all edge costs are uniformly distributed on $[0, 1]$ (note that this can be done probabilistically by a simple Monte Carlo algorithm); Such an algorithm would demonstrate that the corresponding version of MIN VCG PAYMENT is in NP. Another natural question is whether an algorithm that only knows the first k moments of each edge cost distribution can perform reasonably well.

Finally, it would be interesting to generalize our hardness results to the model of [4]: it seems likely that the optimal mechanism in this class is NP-hard to construct, too.

The author would like to thank Amit Sahai and Ken Steiglitz for many useful discussions, and Evdokia Nikolova for helping simplify the proof of Theorem 3.

8. REFERENCES

[1] A. ARCHER AND E. TARDOS, Frugal path mechanisms. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 991–999, 2002

[2] A. BJÖRKLUND, T. HUSFELDT, S. KHANNA, Approximating longest directed paths and cycles. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming*, pages 222–233, 2004

[3] E. CLARKE, Multipart pricing of public goods. *Public Choice*, 8:17–33, 1971

[4] A. CZUMAJ AND A. RONEN, On the expected payment of mechanisms for task allocation. In *Proceedings of the 5th ACM Conference on Electronic Commerce (EC'04)*, 2004

[5] A. CHANDRA, D. HIRSCHBERG, AND C. WONG, Approximate algorithms for some generalized knapsack problems. In *Theoretical Comput. Sci. 3*, pages 293–304, 1976

[6] E. ELKIND, A. SAHAI, AND K. STEIGLITZ, Frugality in path auctions. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 694–702, 2004

[7] J. FEIGENBAUM, C. H. PAPADIMITRIOU, R. SAMI, AND S. SHENKER, A BGP-based mechanism for lowest-cost routing. In *Proceedings of the 21st Symposium on Principles of Distributed Computing*, pages 173–182, 2002

[8] T. GROVES, Incentives in teams. *Econometrica*, 41(4):617–631, 1973

[9] J. HERSHBERGER AND S. SURI, Vickrey prices and shortest paths: what is an edge worth? In *Proceedings of the 42nd Symposium on Foundations of Computer Science*, pages 252–259, 2001

[10] V. KRISHNA, Auction Theory. Academic Press, 2002

[11] D. KARGER, R. MOTWANI, AND G. D. S. RAMKUMAR, On approximating the longest path in a graph. In *Proc. 3rd Workshop on Algorithms and Data Structures*, pages 421–432, 1993

[12] D. KARGER AND E. NIKOLOVA, VCG overpayment in random graphs. Manuscript.

[13] R. MYERSON, Optimal auction design. *Mathematics of Operations Research*, 6:58–73, 1981

[14] M. MIHAIL, C. PAPADIMITRIOU, AND A. SABERI, On certain connectivity properties of the internet topology. In *Proceedings of the 44th Symposium on Foundations of Computer Science*, pages 28–35, 2003

[15] N. NISAN AND A. RONEN, Algorithmic mechanism design. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computation*, pages 129–140, 1999

[16] T. ROUGHGARDEN, Selfish Routing. PhD thesis, Cornell University, Department of Computer Science, May 2002

[17] W. VICKREY, Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16:8–37, 1961